What is claimed is:

1. A method for allowing a profiler to communicate with a virtual machine without regard to a specific implementation of the virtual machine, the method comprising the steps of:
creating one or more heap arenas in a heap; and
using at least one event to dynamically manage storage allocation and storage deallocation in the heap, wherein said at least one event is independent of any algorithm for dynamically managing storage allocation and storage deallocation in the heap.

2. The method of Claim 1, wherein the one or more heap arenas represent one or more logically partitioned portions in the heap.

3. The method of Claim 1, further comprising assigning a unique arena ID to each heap arena.

4. The method of Claim 1, further comprising at least one step of the following steps:
using a new_arena event when a new heap arena is created;
using a delete_arena event with respect to a particular heap arena when all objects within a logically partitioned portion of the heap represented by the particular heap arena have been deleted;
using one or more new_object events when one or more new objects are allocated in the new heap arena;
using one or more delete_object events when one or more dead objects are returned to a free pool in the heap; and
using one or more move_object events when one or more objects are moved from one heap arena to another heap arena.

5. The method of Claim 4, further comprising associating each object with a unique arena ID, a unique object ID, and a unique class ID.

6. The method of Claim 1, further comprising:

initiating a new_arena event to create a new heap arena to represent a logically partitioned portion of the heap;

initiating one or more new_object events when one or more new objects are allocated in the new heap arena ; and

initiating one or more delete_object events when one or more dead objects are returned to a free pool in the heap.

7. The method of Claim 1, further comprising:

initiating a new_arena event to create a new heap arena to represent a logically partitioned portion of the heap;

initiating one or more new_object events when one or more new objects are allocated in the new heap arena;

initiating one or more move_object events when the heap is compacted; and

initiating one or more delete_object events when one or more dead objects are returned to a free pool in the heap.

8. The method of Claim 1, further comprising:

initiating two new_arena events to create a first and second heap arena to represent a corresponding first and second logically partitioned portions in the heap;

initiating one or more new_object events when one or more new objects are allocated in the first heap arena;

initiating one or more move_object events when one or more live objects are moved from the first heap arena to the second heap arena; and

initiating a delete_arena event with respect to the first heap arena when all objects within the first logically partitioned portion of the heap represented by the first heap arena have been deleted.

9. The method of Claim 1, further comprising:

initiating two or more new_arena events to create a plurality of heap arenas to represent a corresponding plurality of logically partitioned portions in the heap;

initiating one or more new_object events when one or more new objects are allocated in a youngest heap arena corresponding to a youngest logically partitioned portion of the heap;

initiating one or more move_object events when one or more objects are moved from a younger heap arena to an older heap arena; and

initiating a delete_arena event with respect to the youngest heap arena when all the objects within the youngest logically partitioned portion of the heap represented by the youngest heap arena have been deleted.

10. A method for interfacing a profiler to a virtual machine, the method comprising the steps of:

receiving from a profiler agent of the profiler at least one of either a request for specific types of events and information or an enablement of notification of specific types of events and information; and

registering the specific types of events and information in which the profiler is interested.

11. A computer-readable medium carrying one or more sequences of one or more instructions for allowing a profiler to communicate with a virtual machine without regard to a specific implementation of the virtual machine, the one or more sequences of one or more instructions including instructions which, when executed by one or more processors, cause the one or more processors to perform the steps of:

creating one or more heap arenas in a heap; and

using at least one event to dynamically manage storage allocation and storage deallocation in the heap, wherein said at least one event is independent of any algorithm for dynamically managing storage allocation and storage deallocation in the heap.

12. The computer-readable medium of Claim 11, wherein the one or more heap arenas represent one or more logically partitioned portions in the heap.

13. The computer-readable medium of Claim 11, further comprising assigning a unique arena ID to each heap arena.

14. The computer-readable medium of Claim 11, further comprising at least one step of the following steps:

using a new_arena event when a new heap arena is created;

using a delete_arena event with respect to a particular heap arena when all objects within a logically partitioned portion of the heap represented by the particular heap arena have been deleted;

using one or more new_object events when one or more new objects are allocated in the new heap arena;

using one or more delete_object events when one or more dead objects are returned to a free pool in the heap; and

using one or more move_object events when one or more objects are moved from one heap arena to another heap arena.

15. The computer-readable medium of Claim 14, further comprising associating each object with a unique arena ID, a unique object ID, and a unique class ID.

16. The computer-readable medium of Claim 11, further comprising:

initiating a new_arena event to create a new heap arena to represent a logically partitioned portion of the heap;

initiating one or more new_object events when one or more new objects are allocated in the new heap arena ; and

initiating one or more delete_object events when one or more dead objects are returned to a free pool in the heap.

17. The computer-readable medium of Claim 11, further comprising:

initiating a new_arena event to create a new heap arena to represent a logically partitioned portion of the heap;

initiating one or more new_object events when one or more new objects are allocated in the new heap arena;

initiating one or more move_object events when the heap is compacted; and

initiating one or more delete_object events when one or more dead objects are returned to a free pool in the heap.

18. The computer-readable medium of Claim 11, further comprising:

initiating two new_arena events to create a first and second heap arena to represent a corresponding first and second logically partitioned portions in the heap;

initiating one or more new_object events when one or more new objects are allocated in the first heap arena;

initiating one or more move_object events when one or more live objects are moved from the first heap arena to the second heap arena; and

initiating a delete_arena event with respect to the first heap arena when all objects within the first logically partitioned portion of the heap represented by the first heap arena have been deleted.

19. The computer-readable medium of Claim 11, further comprising:

initiating two or more new_arena events to create a plurality of heap arenas to represent a corresponding plurality of logically partitioned portions in the heap;

initiating one or more new_object events when one or more new objects are allocated in a youngest heap arena corresponding to a youngest logically partitioned portion of the heap;

initiating one or more move_object events when one or more objects are moved from a younger heap arena to an older heap arena; and

initiating a delete_arena event with respect to the youngest heap arena when all the objects within the youngest logically partitioned portion of the heap represented by the youngest heap arena have been deleted.

20. A computer-readable medium carrying one or more sequences of one or more instructions for interfacing a profiler to a virtual machine, the one or more sequences of one or more instructions including instructions which, when executed by one or more processors, cause the one or more processors to perform the steps of:

receiving from a profiler agent of the profiler at least one of either a request for specific types of events and information or an enablement of notification of specific types of events and information; and

registering the specific types of events and information in which the profiler is interested.

21. A system for profiling a heap, the system comprising:

a memory;

one or more processors coupled to the memory; and

at least one processor configured to:

create one or more heap arenas in a heap; and

use at least one event to dynamically manage storage allocation and storage deallocation in the heap, wherein said at least one event is independent of any algorithm for dynamically managing storage allocation and storage deallocation in the heap.

22. The system of Claim 21, wherein the one or more heap arenas represent one or more logically partitioned portions in the heap.

23. The system of Claim 21, further comprising assigning a unique arena ID to each heap arena.

24. The system of Claim 21, further comprising at least one step of the following steps:

using a new_arena event when a new heap arena is created;

using a delete_arena event with respect to a particular heap arena when all objects within a logically partitioned portion of the heap represented by the particular heap arena have been deleted;

using one or more new_object events when one or more new objects are allocated in the new heap arena;

using one or more delete_object events when one or more dead objects are returned to a free pool in the heap; and

using one or more move_object events when one or more objects are moved from one heap arena to another heap arena.

25. The system of Claim 24, further comprising associating each object with a unique arena ID, a unique object ID, and a unique class ID.

26. The system of Claim 21, further comprising:

initiating a new_arena event to create a new heap arena to represent a logically partitioned portion of the heap;

initiating one or more new_object events when one or more new objects are allocated in the new heap arena ; and

initiating one or more delete_object events when one or more dead objects are returned to a free pool in the heap.

27. The system of Claim 21, further comprising:

initiating a new_arena event to create a new heap arena to represent a logically partitioned portion of the heap;

initiating one or more new_object events when one or more new objects are allocated in the new heap arena;

initiating one or more move_object events when the heap is compacted; and

initiating one or more delete_object events when one or more dead objects are returned to a free pool in the heap.

28. The system of Claim 21, further comprising:

initiating two new_arena events to create a first and second heap arena to represent a corresponding first and second logically partitioned portions in the heap;

initiating one or more new_object events when one or more new objects are allocated in the first heap arena;

initiating one or more move_object events when one or more live objects are moved from the first heap arena to the second heap arena; and

initiating a delete_arena event with respect to the first heap arena when all objects within the first logically partitioned portion of the heap represented by the first heap arena have been deleted.

29. The system of Claim 21, further comprising:

initiating two or more new_arena events to create a plurality of heap arenas to represent a corresponding plurality of logically partitioned portions in the heap;

initiating one or more new_object events when one or more new objects are allocated in a youngest heap arena corresponding to a youngest logically partitioned portion of the heap;

initiating one or more move_object events when one or more objects are moved from a younger heap arena to an older heap arena; and

initiating a delete_arena event with respect to the youngest heap arena when all the objects within the youngest logically partitioned portion of the heap represented by the youngest heap arena have been deleted.

30. A system for profiling a heap, the system comprising:

a memory;

one or more processors coupled to the memory; and

at least one processor configured to:

receive from a profiler agent of a profiler at least one of either a request for specific types of events and information or an enablement of notification of specific types of events and information; and

register the specific types of events and information in which the profiler is interested.

31. A system for profiling a heap, the system comprising:

a storage medium; and

a profiler front-end, wherein the profiler front-end is communicatively coupled to a profiler agent and is on a machine process separate and distinct from the machine process of the profiler agent; and

a profiler agent communicatively coupled to the storage medium, wherein the profiler agent is configured to submit at least one of either a request for specific types of events and information or an enablement of notification of specific types of events and information.

32. A virtual machine comprising a process for creating a plurality of arenas within a heap.

33. A virtual machine having a standard profiler interface accommodating two or more profilers.

34. The virtual machine of Claim 33, wherein one of said two or more profilers reports on activities of a Mark-and-Sweep garbage collector.

35. The virtual machine of Claim 33, wherein one of said two or more profilers reports on activities of a Mark-Sweep-Compact garbage collector.

36. The virtual machine of Claim 33, wherein one of said two or more profilers reports on activities of a Two-Space-Copying garbage collector.

37. The virtual machine of Claim 33, wherein one of said two or more profilers reports on activities of a Generational garbage collector.

38. The virtual machine of Claim 33, wherein one of said two or more profilers reports on activities of a Reference-Counting garbage collector.